# BAB II TINJAUAN PUSTAKA

2.1 Peneliti Terdahulu

### 2.1.1 Peneliti Pertama

Penelitian yang dilakukan oleh Aziz Muliadi (2014) dengan judul “Pemodelan Algoritma Genetika pada Sistem Penjadwalan Perkuliahan Prodi Ilmu Komputer Universitas Lambungmangkurat”. Pada penelitian ini menggunakan metode Algoritma Genetika (AG). Untuk menyelesaikan masalah penjadwalan menggunakan AG dimulai dari pengkodean, menentukan nilai populasi awal, menentukan nilai kromosom secara acak, menentukan nilai fitness untuk meminimalisir jadwal tabrakan, kemudian melakukan seleksi roulettewheel, melakukan pindah silang satu titik potong *(one-point crossever),* kemudian melakukan mutasi pengkodean nilai, elitism. Hasil yang diperoleh dari pemodelan AG pada sistem pendukung keputusan ini merupakan jadwal perkuliahan pada prodi Ilmu Komputer FMIPA Universitas Lambung Mangkurat Banjarmasin.

### 2.1.2 Peneliti Kedua

Penelitian yang dilakukan oleh Via Yisti Vita, dkk (2017) dengan judul “Optimasi Penjadwalan Pegawai Rumah Sakit Menggunakan Algoritma Genetika”. Pada peneliti ini menggunakan metode Algoritma Genetika (AG). Metode AG merupakan algoritma pencarian yang didasarkan atas mekanisme seleksi alami dan evolusi biologis. Hasil dari penelitian ini adalah berupa aplikasi sistem informasi optimasi penjadwalan perawat yang dapat menampilkan informasi data perawat dan ruang serta dapat melakukan proses penjadwalan.

### 2.1.3 Peneliti Ketiga

Penelitian yang dilakukan oleh Ula Yolanda Nailil, dkk (2018) dengan judul “Penjadwalan Dinas Pegawai Menggunakan Algoritma Genetika Pada PT Kereta Api Indonesia (KAI) Daerah Operasi 7 Stasiun Besar Kediri”. Pada penelitian ini menggunakan data jadwal dinas pegawai PT Kereta Api Indonesia Stasiun Besar Kediri tahun 2017, untu mendapatkan hasil penjadwalan yang optimal digunakan metode Algoritma Genetika (AG) sebagai solusi permasalahan berdasarkan aturan yang ditetapkan.

2.2 Teori Terkait

# 2.2.1 Sistem Informasi

Sistem Informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan laporan-laporan yang diperlukan oleh pihak luar tertentu (Sutabri, 2012). Dalam pembuatan sebuah sistem informasi diperlukan adanya perencanaan guna sistem yang dibangun memiliki efisiensi dan efektifitas dalam mengelola sebuah informasi yang ada di sebuah organisasi. Terdapat beberapa tingkatan dalam merencanakan sistem informasi menurut Sutabri (2012), yaitu :

* Tingkat I : Ide, mengetahui perlu adanya perubahan
* Tingkat II : Desain, merancang cara pemecahannya
* Tingkat III : Pelaksanaan, menerapkan desain ke dalam sistem
* Tingkat IV : Kontrol, memeriksa tingkat pelaksanaan dijalankan sesuai dengan *design*
* Tingkat V : Evaluasi, memeriksa apakah perubahan yang terjadi sesuai dengan tujuan semula
* Tingkat VI : Tindak lanjut, melaksanakan perubahan sesuai dengan hasil evaluasi yang ada

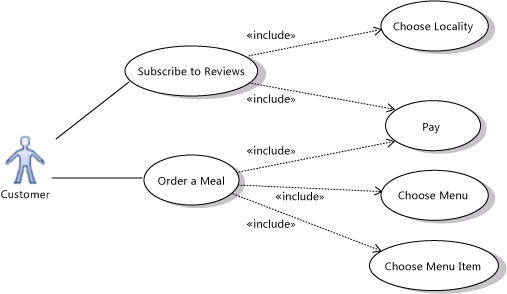
### 2.2.2 Unified Modeling Language

UML(*Unified Modeling Language*) adalah sekumpulan notasi desain yang dengan cepat menjadi standar bahasa desain perangkat lunak. UML menyediakan berbagai kemampuan yang berguna untuk perancang perangkat lunak, termasuk beberapa, tampilan desain yang saling terkait, semantik semiformal yang diekspresikan sebagai model meta UML, dan bahasa terkait untuk mengekspresikan batasan logika formal pada elemen desain. Blok pembangunan utama UML adalah diagram.

Terdapat beberapa macam diagram yang ada dalam UML (Muhamad Muslihudin dkk:2016), diantaranya :

1. Use Case Diagram  : memperlihatkan himpunan use case dan actor-aktor (suatu jenis dari kelas).
2. Sequence Diagram: menekankan pada pengiriman pesan dalam suatu waktu tertentu
3. [Activity Diagram](https://id.wikipedia.org/wiki/Diagram_Aktivitas) : memperlihatkan aliran dari suatu aktivitas lainnya dalam suatu sistem
4. Entity Relationship Diagram : menjelaskan hubungan antar objek-objek data yang mempunyai hubungan.

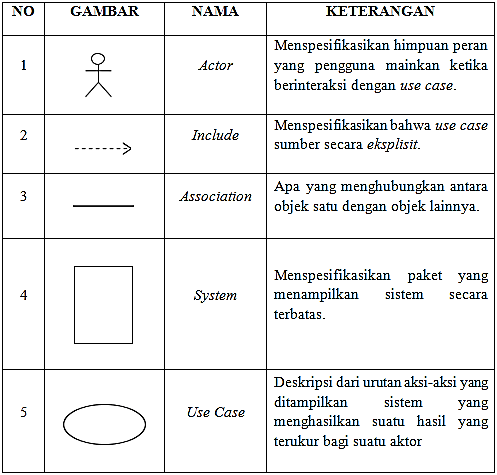
### 2.2.3 Use Case Diagram

 *Use case* adalah sebuah kegiatan yang dilakukan sistem dalam menanggapi permintaan dari pengguna sistem (Triandini Evi:2012).

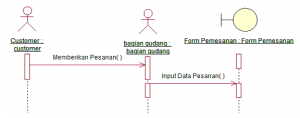
Gambar 2.1 Contoh Use Case Diagram

Diagram *Use Case* berguna dalam tiga hal :

1. Menjelaskan fasilitas yang ada (*requirement*).
2. Komunikasi dengan klien.
3. Membuat tes dari kasus-kasus secara umum.

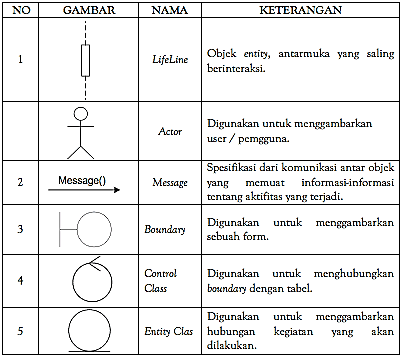
Tabel 2 1Simbol - Simbol Use Case

### 2.2.4 Sequence Diagram

[](http://blogs.unpas.ac.id/ahmadwahyudin/files/2014/12/sequen.png)

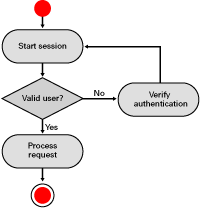
Gambar 2.2 Contoh Sequence Diagram

*Sequence diagram* menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence diagram* adalah gambaran tahap demi tahap yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram* (Siregar Helmi Fauzi dkk:2018)*.*

Tabel 2 2 Penjelasan simbol sequence diagram

### 2.2.5 Activity Diagram

*Activity diagram* menyediakan analis dengan kemampuan untuk memodelkan proses dalam suatu sistem informasi. *Activity diagram* dapat digunakan untuk alur kerja model, *use case individual*, atau logika keputusan yang terkandung dalam metode individual. *Activity diagram* juga menyediakan pendekatan untuk proses pemodelan parallel (Siregar Helmi Fauzi dkk:2018).



Gambar 2.3 Contoh Activity Diagram

Tabel 2 3 Penjelasan tentang activity diagram

#### 

## 2.3 Algoritma

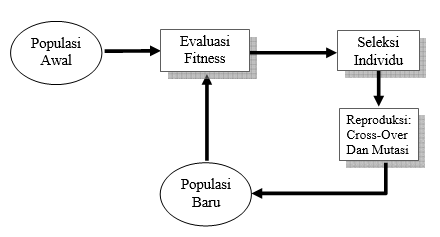
Menurut (Lamhout:2015) algoritma adalah susunan langkah penyelesaian suatu masalah secara sistematis dan logis. Penyajian algoritma secara garis besar dapat dibagi dalam dua bentuk penyajian yaitu tulisan dan gambar. Algoritma yang disajikan dengan tulisan yaitu dengan struktur bahasa tertentu (misalnya bahasa Indonesia atau bahasa Inggris) dan *pseudocode*. *Pseudocode* adalah kode yang mirip dengan kode pemrograman yang sebenarnya seperti Pascal, atau C, sehingga tepat digunakan dalam menggambarkan algoritma yang akan dikomunikasikan kepada *programmer*.

Sedangkan untuk algoritma yang disajikan dengan gambar adalah dengan *flowchart*. *Flowcart* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau merupakan prosedur sistem secara logika. *Flowcart* digunakan untuk alat bantu komunikasi dan untuk dokumentasi.

### 2.3.1 Algoritma Genetika

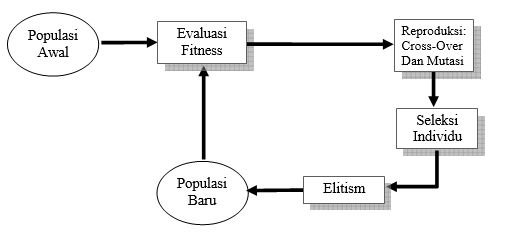
Genetic Algorithm atau Algoritma Genetika (AG) merupakan metode metaheuristic yang terinsipirasi dari proses seleksi natural. GA secara keseluruhan merupakan proses yang terinsipirasi dari proses evolusi biologi yang berdasarkan teori evolusi Charles Darwin (Agusta Gia Muhammad, 2018).Peletak prinsip dasar sekaligus pencipta Algoritma Genetika adalah John Holland. Algoritma Genetika menggunakan analogi secara langsung dari kebiasaan yang alami yaitu seleksi alam. Algoritma ini bekerja dengan sebuah populasi yang terdiri dari individu-individu, yang masing-masing individu mempresentasikan sebuah solusi yang mungkin bagi persoalan yang ada. Dalam kaitan ini, individu dilambangkan dengan sebuah nilai fitness yang akan digunakan untuk mencari solusi terbaik dari persoalan yang ada.

Menurut Muhammad Agusta Gia dalam bukunya berjudul “Algoritma Genetika” terdapat proses umum yang terjadi dalam Algoritma Genetika yaitu *selection*, *crossover* dan *mutation*. Pada organisme proses ini terjadi pada DNA, namun pada AG hal tersebut dikodekan sesuai dengan konteks komputasi yaitu dengan bilangan biner. Setiap faktor atau variabel dari masalah dikodekan dengan biner, dimana setiap sekumpulan biner tersebut disebut individu. Proses evolusi pun biasanya terjadi pada suatu kumpulan individu atau beradaptasinya terhadap lingkungan, kumpulan individu tersebut disebut dengan populasi.

 Maka setiap biner dalam individu dan populasi tersebutlah dilakukan proses *selection, crossover dan mutation*. Output terbaik dari proses evolusi adalah individu yang baik atau dengan sebutan *fitness organism*. Dalam konteks AG, fitness tersebut disebut dengan *fitness value*. Kelebihan dari menggunakan algoritma genetika ini adalah *fitness value* yang dicari dapat menyelesaikan pada masalah global optimum dimana pada metode geometrik lebih sulit ditemukan atau sering terjebak di *local optimum*. Beberapa kasus AG juga dapat mempersingkat waktu dalam pencarian *global optimum* dalam suatu optimasi parameter di metode pengolahan data.

Siklus dari Algoritma Genetika pertama kali dikenalkan oleh David Goldberg,

Gambar 2.4 Siklus Algoritma Genetika oleh David Goldberg

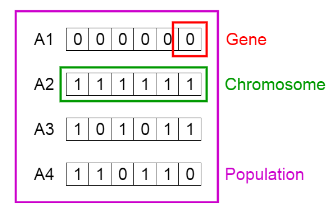
Siklus ini kemudian diperbaiki oleh beberapa ilmuwan yang mengembangkan algoritma genetika, yaitu Zbigniew Michalewicz dengan menambahkan operator elitism dan membalik proses seleksi setelah proses reproduksi.

Gambar 2.5 Siklus Algoritma Genetika oleh Zbigniew Michalewicz

Terdapat 5 fase dalam algoritma genetika, yaitu sebagai berikut (Agusta Gia Muhammad, 2018):

1. Inisialisasi Populasi

Proses dimulai dengan menginisialisasi beberapa individu atau disebut dengan populasi. Setiap individu adalah suatu solusi atau fitness value yang ingin dicari.Setiap individu merupakan sekumpulan dari *Genes* atau gen atau disebut dengan Chromosome (kromosom). Di algoritma genetika, sekumpulan gen ini direpresentasikan dengan kode biner.

 Gambar 2.6 6 Gen, Kromosom dan Populasi

1. *Fitness Function*

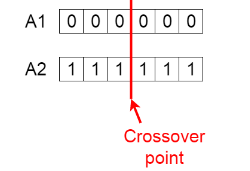
Fungsi fitness menentukan seberapa fit nilai dari suatu individu. Fungsi ini menghasilkan *score* atau *fitness value* yang dicari dari setiap individu.

1. *Selection*

Pada fase ini individu yang memiliki nilai paling fit dibiarkan untuk menjadi *parent* pada generasi berikutnya. Salah satu metode yang paling popular pada fase ini adalah metode roulete. Roulette Wheel Selection dipilih karena proses seleksi menggunakan probabilitas nilai fitness sehingga nilai fitness yang besar memiliki kesempatan untuk terpilih lebih besar dibandingkan dengan nilai fitness yang memiliki nilai lebih kecil.

1. *Cross Over*

Fase ini adalah fase paling signifikan pada algoritma genetika. Setiap pasangan individu yang dijadikan parent kemudian disilangkan untuk membentuk individu baru. Teknik persilangan dingunakan dengan menentukan crossover pointsecara random didalam kromosom.



Gambar 2.7 Crossover point

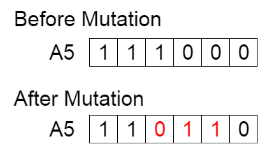
Keturunan akan ‘terlahir’ dengan menukar gen diantara parentsampai titik crossover point.



Gambar 2.8 Penukaran gen antar induk

Setelah penukaran, keturunan baru dimasukan ke populasi.

1. *Mutation*

Setelah keturunan baru dibuat, beberapa gen pada setiap individu dilakukan mutasi dengan probability tertentu, biasanya dengan probabilitas yang rendah. Mutasi dapat dilihat pada gambar 2.9. Mutasi dilakukan untuk mengendalikan keberagaman dalam populasi dan mencegah konvergensi yang terlalu dini.

Gambar 2.9 Sebelum dan Sesudah Mutasi

Algoritma akan berhenti ketika populasi telah konvergen, tidak lagi memproduksi keturunan yang signifikan dari generasi sebelumnya. Dengan kata lain pada tahap ini algoritma genetika telah memberikan solusi dari masalah yang didefinisikan.